Fuzzy Logic in Autonomous Navigation

Alessandro Saffiotti

1 Introduction

The goal of autonomous mobile robotics is to build physical systems that can move purposefully and without human intervention in unmodified environments — that is, in real-world environments that have not been specifically engineered for the robot. The development of techniques for autonomous navigation constitutes one of the major trends in the current research on robotics. This trend is motivated by the current gap between the available technology and the new application demands. On the one hand, the techniques employed in current industrial robots lack the ability to provide flexibility and autonomy: typically, industrial robots perform pre-programmed sequences of operations in highly constrained environments, and are not able to operate in new environments or to face unexpected situations. On the other hand, there is a clear emerging market for truly autonomous robots. Possible applications include intelligent service robots for offices, hospitals, and factory floors; maintenance robots operating in hazardous or inaccessible areas; domestic robots for cleaning or entertainment; autonomous and semi-autonomous vehicles for help to the disabled and the elderly; and so on.

Despite the recent advances in the field of autonomous robotics, many difficulties have to be solved before we can deploy systems that exhibit truly autonomous navigation behavior. Most of the difficulties originate in the nature of real-world, unstructured environment, and in the large uncertainties that are inherent to these environments. First, prior knowledge about the environment is, in general, incomplete, uncertain, and approximate. For example, maps typically omit some details and temporary features, spatial relations between objects may have changed since the map was built, and the metric information may be imprecise and inaccurate. Second, perceptually acquired information is usually unreliable. The limited range, combined with the effect of environmental features (e.g., occlusion) and of adverse observation conditions (e.g., poor lighting), leads to noisy and imprecise data; and errors in the measurement interpretation process may lead to incorrect beliefs. Third, real-world environments typically have complex and unpredictable dynamics: objects can move, other agents can modify the environment, and relatively stable features may change with time (e.g., seasonal variations). Finally, the effect of control actions is not completely reliable: wheels may slip, and a gripper may lose its grasp on an object.

Traditional work in robotics has tried to overcome these difficulties by carefully designing the robot mechanics and sensors, or engineering the environment, or both. This approach is systematically adopted in industrial

In: Fuzzy Logic Techniques for Autonomous Vehicle Navigation.
D. Driankov and A. Saffiotti, eds.
Springer-Physica Verlag, DE, 2001. Pages 3-24.
http://www.aass.oru.se/Agora/FLAR/.

robots, but some amount of environment engineering has often been introduced in autonomous mobile robots as well, from the early days of Shakey until today's service robots that follow a white or magnetic strip on the floor. Engineering the robot or the environment, however, increases costs, reduces robot's autonomy, and cannot be applied to all domains. If we want to build easily available robots that inhabit our homes, offices, or factory floors, we should accept the idea that the platform cannot be overly sophisticated, and that the environment should not be modified. The main challenge of today's autonomous robotics is to build robust control programs that reliably perform complex tasks in spite of the environmental uncertainties.

In this introductory chapter, we shall hint at some of the ways in which fuzzy logic can be used, and has been used, to address this challenge. In the next section, we discuss the architectural solutions devised to face the challenge, and introduce a framework that helps us to separate its different sub-problems. In the following sections, we discuss four of these sub-problems which provide the framework for this volume: (i) how to design robust behavior-producing modules; (ii) how to coordinate the activity of several such modules; (iii) how to use data from the sensors; and (iv) how to integrate high-level reasoning and low-level execution. We then conclude the chapter with a few comments on the pros and cons of using fuzzy logic for autonomous robot navigation.

2 The challenge of autonomous navigation

Any approach to control a dynamic system needs to use some knowledge, or *model*, of the system to be controlled. In the case of a robot, this system consists of the robot itself *plus* the environment in which it operates. Unfortunately, while a model of the robot on its own can normally be obtained, the situation is different if we consider a robot embedded in the type of real-world, non-engineered environments which we would like to consider. As noticed above, these environments are characterized by the ubiquitous presence of uncertainty; more annoyingly, the nature of the involved phenomena is such that we are often not able to precisely model or quantify this uncertainty.

Consider, for example, the uncertainty induced by the presence of people. People move around, and they may change the position of objects and furniture; in most cases, however, we cannot hope to write, say, a meaningful probability distribution that characterizes these events. The interaction of the robot with the environment causes similar difficulties: the results of the robot's movement and sensing actions are influenced by a number of environmental conditions which are hard to be accounted for. For example, the error in the robot's motion may change as a result of a wet floor; the quality of visual recognition may deteriorate on a Spring day with rapidly changing lighting conditions; and the reliability of distance measured by a sonar sensor

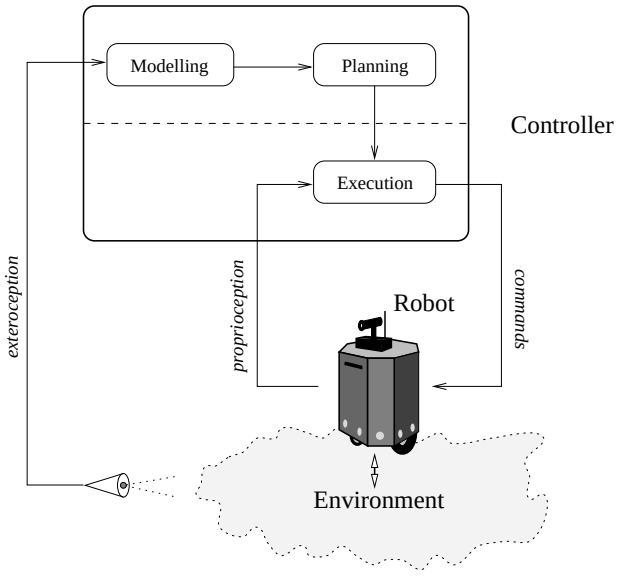

Fig. 1. Hierarchical architecture. The high-level layer builds a model of the environment and generates a plan for action. The low-level blindly executes this plan.

is influenced by the geometry and the reflectance properties of the objects in the environment.

A common strategy to cope with this large amount of uncertainty is to abandon the idea of completely modeling the environment at the design phase, and endow the robot with the capability of building this model by itself on-line. This strategy leads to the so-called *hierarchical* architectures, sketched in Figure 1.[1] The robot uses *exteroceptive* sensors, like a camera or a sonar sensor, to observe the state of the environment; it uses *proprioceptive* sensors, like a compass or shaft encoders on the wheels, to monitor the state of its own body parts. (Although exteroceptive sensors are physically mounted on the robot, we draw them as a separate entity to emphasize the difference between exteroceptive and proprioceptive information.) By using the exteroceptive sensors, the robot acquires a model of the workspace as it is at the moment when the task must be performed. From this model, a planning program builds a plan that will perform the given task in the given environment. This plan is then passed to a lower-level control program for execution. Typically, execution proceeds "blindly" — the controller may use a model of the robot and monitor the state of the robot's effectors (propri-

[1] The term "hierarchical" is overloaded and inevitably ambiguous. It is used here to refer to a control hierarchy where the higher-level decides the set-points to be achieved by the lower-level. Some authors prefer the term "functional decomposition" to identify this type of architecture [6].

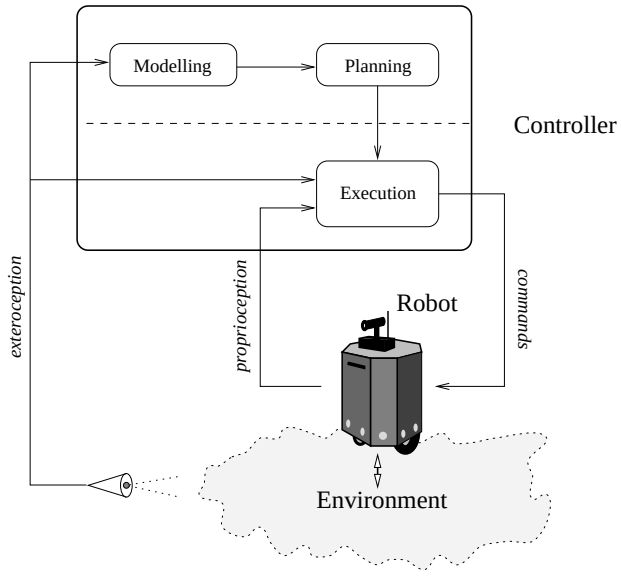

Fig. 2. Hybrid architecture. The lower layer uses perception to dynamically adapt plan execution to the environmental contingencies. The execution module must simultaneously consider demands coming from the plan and from the environment.

oception), but it does not try to sense or model the environment anymore. The Modeling and Planning modules thus take care of analyzing the execution environment and generating a control program (plan) for that specific environment, to be executed by the controller. In a sense, the hierarchical approach factors the environment out of the controlled system, making the control problem tractable. This approach has been extensively used in the robotics literature; in most cases, the plan consists of a path leading to the goal, and execution consists in tracking this path.

It is not difficult to see the limitations of the hierarchical approach when dealing with real-world environments. The model acquired by the robot is necessarily incomplete and inexact, due to the uncertainty in perception. Moreover, this model is likely to rapidly become out of date in a dynamic environment, and the plan built from this model will then turn out to be inadequate to the environment actually encountered during execution. The fact that the modeling and planning processes are usually computationally complex and time consuming exacerbates this problem: intuitively, the feedback loop with the environment must pass through all these processes — for this reason, this approach is also known as the "Sense-Model-Plan-Act", or SMPA approach. The complexity of the processes in the SMPA loop can easily make the response time of the robotic system of the order of seconds, far too much for dynamic environments.

By the end of the eighties technological improvements had caused the cost of mobile platforms and sensors to drop, and mobile robots began to appear in several AI research labs. Research on autonomous navigation was strongly pushed, and a number of new architectures were developed that tried to integrate perception and action more tightly. The general feeling was that planning should make as few assumptions as possible about the environment actually encountered during execution; and that execution should be sensitive to the environment, and adapt to the contingencies encountered. To achieve this, perceptual data had to be included into the executive layer, as shown in Figure 2.

This apparently simple extension has two important consequences. First, it makes robot's interaction with the environment much tighter, since the environment is now included in a closed-loop with the (usually fast) execution layer. Second, the complexity of the execution layer has to be greatly increased, since this needs now to consider multiple objectives: pursuing the tactical goals coming from the planner, and reacting to the environmental events detected by perception.

Most researchers have chosen to cope with this complexity by a *divide and conquer* strategy, but they advocate different styles of divisions. Some have stayed within the hierarchical tradition, and have further split the execution layer into a sequencing layer and process layer. In the so-called "3-tier approach," the sequencing layer generates set-points by considering both tactical goals and perceptual events, and the process layer controls the effectors to achieve these set-points [10]. Others have departed from the hierarchical approach, and have decomposed the execution layer into a set of small independent decision-making units, or *behaviors* [1,6]. In the so-called "behavior-based approach," each behavior fully implements a control policy for one specific sub-task, like following a path, avoiding sensed obstacles, or crossing a door. Simple behaviors are then combined in order to produce a complex strategy able to pursue the strategic goals of the agent, while effectively reacting to contingencies. Most current autonomous robots are based on a *hybrid architecture* like the one sketched in Figure 2 in which the execution layer incorporates elements of both the 3-tier and the behavior-based approach.

Hybrid architectures do not solve the autonomous navigation problem, but they provide a convenient framework in which the different sub-problems can be dealt with and integrated. In the rest of this introductory chapter, we focus on four sub-problems which: (i) play an important role in autonomous navigation; and (ii) are particularly prone to solutions based on fuzzy logic.

3 Fuzzy logic for behavior design

The first problem that we discuss is the need to design simple individual behaviors that guarantee robust operation in the presence of uncertainty. A typical example is the design of an obstacle avoidance behavior that is effec-

tive in face of unforeseen obstacle configurations. Although many solutions have already been reported in the literature, the continuing development of new proposals suggests that this issue has not settled down yet.

Perhaps the most classical type of behavior used in mobile robots is path tracking: the controller is given a path in some internal reference frame, and it generates motor commands in order to follow it as closely as possible. Path tracking may be surprisingly difficult. The kinematics and dynamics of the robot may be complex and non-linear, and the interaction between the vehicle and the terrain may be hard to model in general. These characteristics led several authors to use fuzzy control techniques for path tracking. In this volume, the chapter by Ollero *et al.* [27] shows two ways to apply fuzzy control techniques to the tracking of arbitrary continuous paths. In the first one, the fuzzy controller directly generates the robot's controls; in the second one, it generates the motion parameters used by a geometric method. Both have been tested on a real robot with good results.

Tracking a precomputed path is an effective way to bring the robot to a target position when two conditions are verified: the assumptions used during the computation of the path are still valid at execution time (e.g., the environment was correctly modeled, and it has not changed afterwards); and the robot is able to reliably establish its position with respect to the path. These conditions are rarely met in real world environments, and many researchers have preferred to equip their robots with sensor-based behaviors. A sensor-based behavior implements a control policy based on external sensing; intuitively, the robot moves with respect to features in the environment, rather than with respect to an internally represented path. Typical examples are moving along a wall, reaching a light source, and avoiding obstacles.

The first reported uses of fuzzy control in mobile robotics belong to this class. For example, in 1985 Sugeno and Nishida developed a fuzzy controller able to drive a model car along a track delimited by two walls [39]. Shortly after, Takeuchi *et al.* [41] developed a fuzzy controller for obstacle avoidance based on a simple algorithm to obtain information about occupied and free areas in front of the robot from a video camera. Still today, most of the sensor-based behaviors found in mobile robots implement the same two fundamental tasks as above: following environmental features (walls, road edges, white lines on the floor, or other), and avoiding obstacles. In this volume, the chapters by Godjevac and Steele [12] and by Hoffmann [15] show how behaviors of this type can be obtained by a combination of fuzzy logic and machine learning.

More extensively developed autonomous robots have been equipped with rich suites of behaviors, covering all the elementary sub-tasks that they need to perform. For example, the systems described in the chapters by Pin and Watanabe [29], by Goodridge and Kay [13], by Tunstel [42], and by Surmann and Peters [40], include fuzzy behaviors for going to a given position, for orienting towards a target, for docking to an object, for crossing a door, and so

on. The chapter by Murphy [26] illustrates a different type of behavior, called "tactical." A tactical behavior acts as a supervisory controller, observing the recent execution in order to modify some navigation parameters; the authors report on a tactical fuzzy behavior for speed regulation based on the amount of recent turning, and on the density of obstacles.

All the behaviors discussed above can be classified as *basic*, in that they only care for one single objective, or performance criterion—for example, a path tracking behavior does not account for the possibility of an obstacle blocking the path. To perform autonomous navigation, however, a robot needs the ability to exhibit *complex* behaviors, that take multiple objectives into account: for example, following a given path while avoiding unforeseen obstacles in real time.

Fuzzy controllers are typically designed to consider one single goal. If we want to consider two (or several) interacting goals, we have two options. We can write one set of complex rules whose antecedents consider both goals simultaneously; or we can write two sets of simpler rules, one specific to each goal, and combine their outputs in some way. For example, if the two goals are tracking a path and avoiding obstacles, the first approach consists in using fuzzy rules of the general form

$$\begin{cases} \text{IF } \textit{track-condition}_1 \text{ AND } \textit{avoid-condition}_1 \text{ THEN } \textit{command}_{11} \\ \text{IF } \textit{track-condition}_1 \text{ AND } \textit{avoid-condition}_2 \text{ THEN } \textit{command}_{12} \\ \vdots \\ \text{IF } \textit{track-condition}_n \text{ AND } \textit{avoid-condition}_m \text{ THEN } \textit{command}_{nm} \end{cases}$$

while the second approach would use fuzzy rules of the general form

$$\begin{cases} \text{IF } \textit{track-condition}_1 \text{ THEN } \textit{track-command}_1 \\ \vdots \\ \text{IF } \textit{track-condition}_n \text{ THEN } \textit{track-command}_n \end{cases}$$

$$\begin{cases} \text{IF } \textit{avoid-condition}_1 \text{ THEN } \textit{avoid-command}_1 \\ \vdots \\ \text{IF } \textit{avoid-condition}_m \text{ THEN } \textit{avoid-command}_m. \end{cases}$$

The second approach is similar in spirit to the behavior-based approach, where the overall control problem is decomposed into small behaviors, each one focusing on only a small portion of the input space. The chapter by Goodridge and Kay [13] in this volume discusses the behavior-based approach from the perspective of reducing complexity. Arguably the most difficult problem with this approach is how to fuse the commands issued by the different rule-sets. We shall come back to this issue shortly.

Whether a complex behavior is better implemented by a monolithic or a partitioned set of rules is a difficult question. The monolithic solution can take better care of the interactions between the goals, and should be preferred

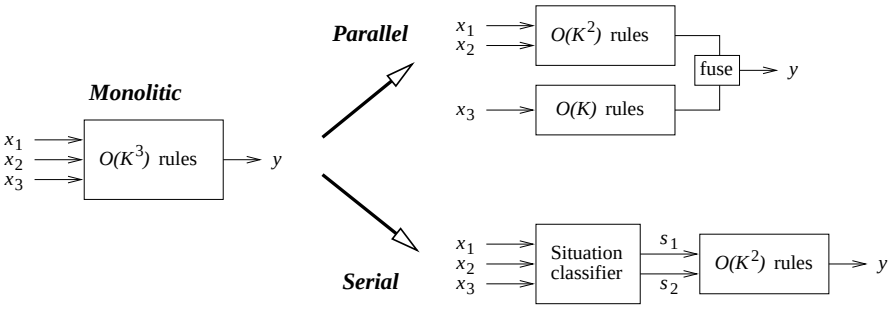

Fig. 3. Parallel and serial decomposition of a complex behavior.

when these interactions are important. Unfortunately, the monolithic solution can easily become intractable, as the number of rules tends to grow exponentially in the size of the input space: in the above example, the monolithic approach needs nm rules, as opposed to the $n + m$ rules of the partitioned approach.

Another possibility to reduce the complexity of the input space is to introduce a limited number of intermediate variables, meant to classify the different "perceptual situations" that are relevant to the robot's behavior. The input from the sensors is fed to a situation classification module that sets the values of these intermediate variables; typical intermediate variables can be "facing_obstacle" and "wheels_skidding." These variables are then used by the fuzzy control rules. Figure 3 compares this serial decomposition with the parallel decomposition of a complex behavior.

Early examples of serial decompositions are provided by Maeda *et al.* [24], who use fuzzy rules to classify the shape of a road from image data; and by Altrock *et al.* [43], who use fuzzy rules to diagnose situations in which their model car is sliding or skidding. The latter authors note that situation clustering has made a difficult problem (the dynamic stabilization of a model car at high speed) manageable. In this volume, this approach is illustrated in the chapters by Pin and Watanabe [29], who collapse the readings from 24 sonars into three variables "left", "center" and "right"; by Surmann and Peters [40], who distinguish between 21 different navigation situations; and by Zhang and Knoll [48], who use situation evaluation as a way to modularize controllers.

4 Fuzzy logic for behavior coordination

Since the first appearance of behavior-based approaches in the mid-eighties, authors have noticed the importance of the problem of behavior coordination: how to coordinate the simultaneous activity of several independent behavior-producing units to obtain an overall coherent behavior that achieves the

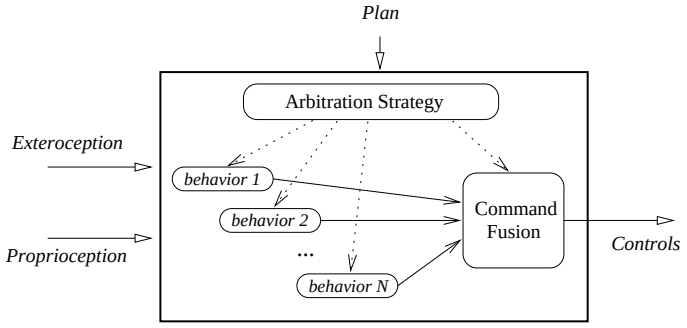

Fig. 4. Behavior-based organization of the execution module. Complexity is managed by a divide and conquer strategy.

intended task. Still today, the problem of behavior coordination is generally recognized as being the Achilles' heel of behavior-based robotics.

As suggested by Fig. 4, we can split the behavior coordination problem into two conceptually different problems: (i) how to decide which behavior(s) should be activated at each moment — and, possibly, how much so; and (ii) how to combine the results from different behaviors into one command to be sent to the robot's effectors — possibly, taking weights into account. We call these the *behavior arbitration* and the *command fusion* problem, respectively.

The arbitration policy determines which behavior(s) should influence the operation of the robot at each moment, and thus ultimately determines the task actually performed by the robot. Early solutions, like the subsumption architecture proposed by Brooks [6], relied on a *static* arbitration policy, hard-wired into a network of suppression and inhibition links. Most current architectures, however, use *dynamic* arbitration schemes where the decision of which behavior to activate depends on the current plan and on the environmental contingencies.

Fuzzy logic has been used to implement both static and dynamic arbitration schemes. Typically, the rationale for using a fuzzy arbitration scheme is the ability to encode the partial activation of behaviors, and to perform a smooth transition between behaviors. The first example of a static (fixed-priority) fuzzy arbitration scheme is probably due to Berenji *et al.* [3], who applied it to the classical cart-pole control problem. In this volume, the chapter by Pin and Watanabe [29] proposes a static arbitration scheme based on the suppression and inhibition mechanisms used in subsumption architectures, but generalized to operate on fuzzy behaviors. Dynamic arbitration schemes can be obtained using fuzzy meta-rules, or *context rules*, of the form

IF *context* THEN *behavior*,

meaning that *behavior* should be activated with a strength given by the truth value of *context*, a formula in fuzzy logic. Fuzzy context rules have

been initially applied by Sugeno *et al.* [38] to switch between flight modes in a fuzzy-controlled unmanned helicopter. In most current instances, fuzzy context rules are used to arbitrate behaviors which are themselves implemented by fuzzy logic, although this need not be the case (e.g., [11,28]).

Turning now our attention to command fusion, perhaps the most popular approaches are those based on the so-called artificial potential fields [17,21]. These approaches use a vector summation scheme: the output of each behavior is represented by a force vector, and outputs from different behaviors are combined by vector summation. The robot "responds" to the force resulting from the combined vector. In a sense, the individual decisions of different behaviors are fused into a tradeoff, combined decision.

The "fuzzy logic" view of command fusion is due to Ruspini [34], and is grounded in the interpretation of fuzzy logic as a logic of graded preferences. While in potential field approaches each behavior produces an individual *decision*, represented by a force vector, in the fuzzy logic view each behavior is seen as an agent expressing *preferences* as to which command to apply. These preferences are represented by a possibility distribution (or a fuzzy set) over the command space. We can then use fuzzy operators to combine the preferences of different behaviors into a collective preference, and finally chose a command from this collective preference. (Performing this choice corresponds to the defuzzification step in fuzzy controllers.) Curiously enough, a similar scheme was also suggested, in a naive form, by two roboticists who were unaware of fuzzy logic but were frustrated by the pitfalls of the existing arbitration schemes [31]. In this volume, the chapter by Pirjanian and Matarić [30] relates fuzzy command fusion to multiple objective decision making.

It is important to note that the decision taken from the collective preference can be different from the result of combining the individual decisions. Each individual decision issued by a behavior tells us which is *the* preferred command according to that behavior, but does not tell anything about the desirability of alternatives. Preferences contain more information, as they give a measure of desirability for each possible command: combining preferences thus uses more information than combining vectors, and can produce a different final decision. Fig. 5 graphically illustrates this point in the case of two behaviors $B1$ and $B2$ both controlling the steering angle. $B2$ has a narrow preference which is mostly included in the preference expressed by $B1$. The decision taken from the common preference (b) is close to the decision that would be taken by $B2$ alone, and also satisfies $B1$. By contrast, combining individual decisions (a) results in an intermediate value (4.0) which does not satisfy $B2$. What this argument shows is that fuzzy command fusion is fundamentally different from vector summation, i.e., from potential field approaches, and that it may produce decisions that better satisfy all of the contributing behaviors.

Most proponent of fuzzy command fusion have considered the problems that may arise by blindly applying defuzzification to the combined fuzzy

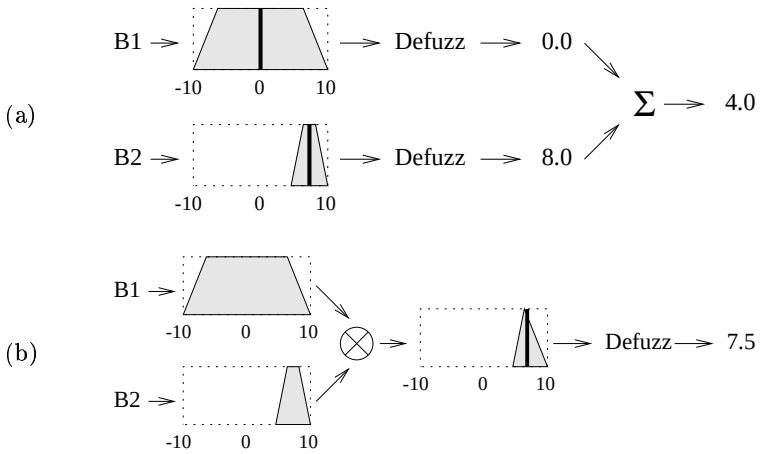

Fig. 5. Two approaches to command fusion: (a) combining individual decisions; (b) combining individual preferences. The final result may be different.

preference set. In particular, when this set is not unimodal defuzzification may result in the selection of an "undesirable" control value, i.e., a value which lays in the gap between two peaks of the combined set, and has low membership in this set. In the case of robot control, this may mean that the robot, having the option of avoiding an obstacle from the right or from the left, decides to go straight. Typically, three types of attitudes toward this problem are found in the literature:

- design a "better" defuzzification operator;
- enforce some consistency criteria on the designer of the fuzzy rule set, so to prevent the generation of inconsistent or ambiguous preferences; and
- report the occurrence of undesirable defuzzified values to the higher-layers, which are responsible for analyzing the problem and breaking the tie.

Examples of each approach can be found in this volume in the chapters by Pirjanian and Matarić [30], by Pin and Watanabe [29], and by Goodridge and Kay [13], respectively.

The mechanism provided by fuzzy logic to realize behavior arbitration and command fusion can be combined by using: (i) fuzzy context rules to express an arbitration policy, and (ii) fuzzy preference combination to perform command fusion. For instance, we can use the following context rules to navigate to a target while reactively avoiding obstacles on the way:

IF obstacle-close THEN Avoid-Obstacle
IF ¬(obstacle-close) THEN Go-To-Target.

When the obstacle is only partially close, both behaviors are partially activated, and fuzzy command fusion is used to select control values that steer away from the obstacle while keeping the generic goal direction.

This general scheme for behavior coordination was initially suggested by Ruspini [32], and fully developed by Saffiotti *et al.* [35,36] under the name of *context-dependent blending* of behaviors, or CDB. CDB is currently used in several autonomous robots. In this volume, the chapters by Goodridge and Kay [13], by Tunstel [42], and by Surmann and Peters [40] describe robotic systems that use some form of CDB. Other examples of system that use CDB-like arbitration schemes can be found in [2,25,28,44,46]. In some cases, CDB has been implemented in a hierarchical fuzzy controller [19]. The hierarchical organization answers the criticism of non-scalability sometimes directed at fuzzy control techniques [4].

5 Fuzzy logic for environment modeling

Autonomous robots need to use perceptual information in order to reduce uncertainty. The issue of artificial perception is amazingly vast, and we only consider here one specific instance which is particularly relevant to the problem of autonomous robot navigation, and for which solutions based on fuzzy logic have been proposed: the use of sensor data and prior knowledge to build and maintain a global model of the environment, or *map*.

Environment information can be represented at several levels of abstraction, ranging from a purely qualitative representation of major objects (e.g., rooms in a building) and of their topological connections, all the way down to a fully detailed geometrical representation of the environment. A common belief in the robotics field is that robots need to represent and reason about information at different levels of abstraction [20]. There are several reasons for this. First is epistemic adequacy: different tasks call for different types of representation. For example, global navigation strategies are more easily planned using a topological map, on which we can decide the sequence of rooms and corridors to be traversed; but fine motion control needs geometric information to precisely control navigation among features and obstacles. Second is computational adequacy: geometric information is difficult to collect and expensive to handle, and we cannot pay the price to maintain a detailed geometric representation of the entire environment where the robot can operate. Finally, there is ontological adequacy: fine grained information is difficult to obtain *a priori* and is likely to change with time; coarse maps are easier to obtain and more prone to remain valid over time.

Another important aspect of a map representation is the way in which it can account for the uncertainty in its properties. Most of the approaches in the robotic domain are based on a probabilistic representation of uncertainty. A probability-based representation is adequate when two conditions hold: (i) the underlying uncertainty can be given a probabilistic interpretation; and

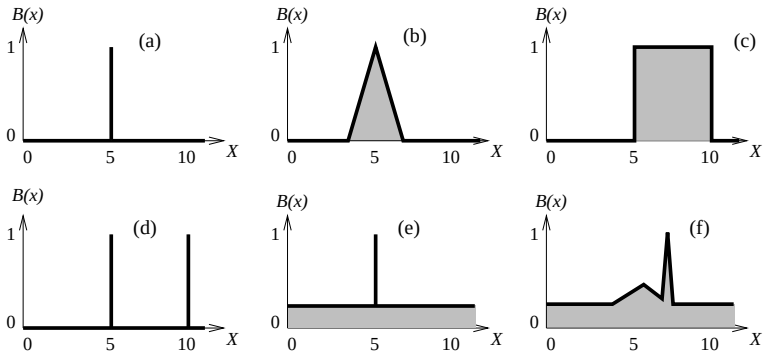

Fig. 6. Representing different types of uncertainty by fuzzy sets: (a) crisp; (b) vague; (c) imprecise; (d) ambiguous; (e) unreliable; (f) mixed. (Adapted from [37].)

(ii) the data required by probabilistic techniques is available. Both conditions may be violated in the case of autonomous robots, and techniques based on fuzzy set theory may then offer a valuable alternative.

Fuzzy subsets of the given space can be used to represent the approximate location of an object, under a possibilistic reading [47]. If B is a fuzzy set representing the approximate location of object b in one dimension, then we read the value of $B(x) \in [0,1]$ as the *degree of possibility* that object b be actually located at x. This representation allows us to model different aspects of locational uncertainty. Figure 6 shows six approximate locations: (a) is a crisp (certain) location; in (b), we know that the object is located at approximately 5 (this is commonly referred to as "vagueness"); in (c), it can possibly be located anywhere between 5 and 10 ("imprecision"); in (d), it can be either at 5 or at 10 ("ambiguity"); (e) shows a case of "unreliability": we are told that the object is at 5, but the source may be lying, and there is a small "bias" of possibility that it can be located just anywhere. (As an extreme case, we represent total ignorance by the "vacuous" location $P(x) = 1$ for all x: any location is perfectly possible.) Finally, (f) combines vagueness, ambiguity and unreliability. The information provided by a measurement device can present any of the above aspects, alone or in combination. It is important to emphasize that a degree of possibility is *not* a probability value: in particular, there is no necessary relation between the observed frequency of a location and its possibility, and degrees of possibility of disjoint locations need not add up to one.

The chapters in the third part of this volume present three approaches to map building based on fuzzy sets. The chapter by Fabrizi *et al.* [8] proposes a variant of Moravec and Elfes' occupancy grids in which each cell c is associated with a possibility distribution π_c over the set $\{\mathsf{occupied}, \mathsf{free}\}$ (or, equivalently, the sets of free and occupied cells are fuzzy sets). The authors

give several reasons that justify the use of fuzzy sets to represent occupancy information.

The chapters by López-Sánchez *et al.* [23] and the one by Gasós [9] use similar feature based representations in which the contours of objects are represented by fuzzy segments, i.e., segments whose width and length are trapezoidal fuzzy sets. The former chapter focuses on cooperative exploration by a team of homogeneous robots, while the latter focuses on the integration of linguistic descriptions with a sonar-based map.

Gasós' contribution also brings about another fundamental issue in environment modeling: in order to make effective use of a map, the robot must be able to estimate its own position with respect to this map. This is usually referred to as the *self-localization* problem. Most existing approaches to self-localization are based on a probabilistic representation of spatial uncertainty. These approaches can be very effective, provided that the required data is available, that is, that: (a) we have an accurate dynamic model of the robot, (b) we have an accurate stochastic model of the sensors, and (c) these systems do not change in unpredictable ways with time.

These conditions are pretty demanding, and they may easily be violated in the case of autonomous robots. When this happens, fuzzy logic may offer valuable alternatives which require less demanding assumptions: for example, fuzzy-based localization methods only need qualitative models of the system and of the sensors. Gasós' technique performs self-localization by comparing the partial fuzzy map built by the robot during navigation with a pre-existing global map, usually built in a preceding exploration phase. The algorithm has shown good performance even in situations of high uncertainty, when so-called *global localization* must be performed.

6 Fuzzy logic for layer integration

The last issue that we discuss is the problem of how to link, or *integrate*, the different levels of representation and reasoning that must be present in an autonomous agent. The study of the link between abstract cognition and physical activity has been attracting attention from philosophers and scientists for centuries. In robotics, the integration problem mainly consists in connecting the low-level and high-level layers in the hybrid architecture sketched in Figure 2 — more precisely, it consists in maintaining the right correspondence between the execution of behaviors at the lower level, and the achievement of the goals considered by the higher level.

As noted by Hanks and Firby [14], an essential aspect of the integration problem is the definition of a *shared plan representation*, that is, a representation that can be easily manipulated and reasoned about by the high-level processes, and that can also be effectively used by the low-level processes to control execution. Unfortunately, the definition of a shared plan representation is complicated by the fact that the processes and the representations

IF obstacle	THEN AVOID
IF (¬obstacle ∧ in(Corr-1) ∧ ¬in(Corr-2))	THEN FOLLOW(Corr-1)
IF (¬obstacle ∧ in(Corr-2) ∧ ¬near(Door-5))	THEN FOLLOW(Corr-2)
IF (¬obstacle ∧ near(Door-5) ∧ ¬in(Room-5))	THEN CROSS(Door-5)
IF in(Room-5)	THEN STILL

Fig. 7. A set of fuzzy situation-action rules can represent a full plan. If executed in the right environment, this plan brings the robot into room-5. (Adapted from [35])

used at the different layers are essentially different. For example, high-level reasoning processes typically manipulate symbolic representations that are based on an abstract model of reality, while low-level processes manipulate numerical data that are grounded in the physical world through the robot's sensors and effectors. These two levels call for different types of tools, and pose different requirements on plan representation. For example, many robot execution systems are based on plans written in elaborated reactive languages (e.g., [7,14]), which are too complex to be generated by current planners.

Fuzzy logic has an extremely attractive feature in this respect: its ability to represent both the symbolic and the numeric aspects of reasoning. Fuzzy logic can be embedded in a full logical formalism, endowed with a symbolic reasoning mechanism; but it is also capable of representing and processing numerical data. Hence, fuzzy logic can be used as a tool to represent plans that can be shared between the higher and lower level. To see this, recall that fuzzy rules can be used to express complex arbitration strategies. These strategies can be seen as specifications of plans of action for the robot, in the form of *situation* → *action* rules. For example, the set of context rules listed in Figure 7 constitutes a plan to reach room 5 as follows: when in corridor-1, the robot follows it until it reaches corridor-2; then it follows corridor-2 until it gets close to door-5; and so on. The plan also incorporates an avoidance behavior to go around possible obstacles. If executed in an environment with the right topological relations, this plan will get the robot into room-5 from anywhere in corridors 1 and 2.

Plans expressed by fuzzy context rules satisfy the requirements for a shared plan representation: they have a simple and logical format which makes them suitable to be automatically generated by symbolic processes; and they can be executed by a (hierarchical) fuzzy controller via the CDB mechanism. Plans representations of this type have been proposed by Saffiotti *et al.* [35], who rely on a standard goal regression planner to generate navigation plans for the robot FLAKEY. Two additional examples are presented in the last part of this volume. The chapter by Surmann and Peters [40] shows the use of this type of plans on the robot MORIA. The chapter by Zhang and Knoll [48] shows the use of fuzzy rules to implement a simple reactive planner that decides the need to re-plan, and that arbitrates between goal reaching and obstacle avoidance.

An important advantage of using a uniform formalism across different layers is the possibility to formally analyze the behavior of the resulting, multilevel system as a whole. For example, Wang [45] proposes a mathematical analysis of a three-level architecture where the bottom level is designed using standard control theory techniques, while fuzzy rules are used at the two upper levels. And Saffiotti *et al.* [35] prove a few composition theorems that relate CDB composition of behaviors to the achievement of composite goals, thus linking the goal decomposition performed by a planner to the behavior composition performed by a CDB-based fuzzy executor.

7 Grandeur et misère of fuzzy logic

Fuzzy logic has found a number of successful applications in the domain of autonomous robot navigation. The earliest one, and still the most common, is the use of fuzzy control to implement individual behavior units. In general, authors have noted four main advantages of using fuzzy control in this domain. First, the fuzzy rule format and the notion of linguistic variable make it easy to write simple and effective behaviors for a variety of tasks without the need to rely on a precise mathematical model. Second, the interpolation mechanism implemented by fuzzy controllers results in smooth motion of the robot, and in graceful degradation in face of errors in sensor data and of fluctuations in the system parameters. Third, thanks to their qualitative nature, fuzzy rule-based behaviors are prone to be transfered from one platform to another with few modifications. Finally, fuzzy controllers lend themselves to efficient implementations, including hardware solutions. The contributions collected in this volume illustrate all of these points.

While the possibility to write effective controllers without using explicit mathematical models constitutes a strength of the fuzzy control technology, it is also the source of its two main drawbacks. First, not having a mathematical model, we cannot use classical tools for formal design. The design of fuzzy controllers is typically done by eliciting the fuzzy rules and the membership functions from a human who knows how to control the system. Debugging and tuning then proceeds by trials and errors. The second, related drawback is that once we have the controller we cannot have any guarantee that it will produce the desired behavior other than by performing empirical tests. While some tools exist to prove, say, stability of a fuzzy controller, these tools rely on the availability of a model of the system (e.g., [18]). Whether or not the heuristic approach provided by fuzzy control is more effective than trying to build a suitable analytical model of the system depends on the specific domain and task. In the autonomous robotics domain, where a mathematical model of the environment in which the robot must operate is usually not available, fuzzy control often turns out to be an attractive option. An additional possibility is to use learning technique to elicit the

rules and membership functions of a fuzzy controller: several contributions in this volume illustrate this possibility.

The basic idea of fuzzy controller, to use linguistic rules to smoothly switch between different control regimes, naturally generalizes to the idea of using fuzzy context rules to arbitrate between different behavior. Like fuzzy control rules, context rules allow us to write complex arbitration strategies in a modular way using a logical format. The fact that the same format is used for the control rules and the arbitration rules makes it easy to write increasingly complex behaviors in a hierarchical fashion. Fuzzy command fusion can then be used to combine recommendations from concurrent behaviors. The resulting scheme, called Context Dependent Blending (CDB), gives us a powerful tool for writing complex behavior coordination strategies. This scheme subsume other coordination schemes commonly used in robotics, including behavior switching and (weighted) vector summation. However, CDB does not give us a solution to the general problem of behavior coordination. For example, we do not know how to distinguish a situation in which different commands proposed by different behaviors should be averaged, from one in which these different commands should be regarded as a sign of a conflict which ought to be resolved in some way. These problems are inherent to any form of local combination, and can be seen as instances of the general problem of relating local computation (or action) to global results (or goals). As such, these problems can only be solved by a careful integration between local and global reasoning — that is, between the higher and lower layers of a hybrid control architecture.

One of the problems that complicate the integration between different layers is that these layers rely on essentially different types of representation and reasoning tools. Typically, the higher layers perform global computations on symbolic representations, while the lower layers perform local computations on numerical representations. To this respect, fuzzy logic appears to be an extremely interesting tool to address the layer integration problem thanks to its intrinsic ability to integrate numeric ("fuzzy") and symbolic ("logic") aspects of reasoning. Curiously enough, this potential of fuzzy logic seems to have been scarcely noticed until now, only few works have been reported in the literature that use fuzzy logic to address the integration problem. Given the key role played by integration issues in autonomous robotics, we speculate that the hybrid symbolic-numeric nature of fuzzy logic will be pivotal to its future exploitation in this domain.

In addition to the ability of writing complex control laws, fuzzy sets and fuzzy logic offer a powerful tool for representing the type of knowledge encountered in the modeling of real world environments. Fuzzy sets include crisp sets as a special case, hence they can represent precise and complete information whenever it is available. But they are also prone to represent uncertain, vague, and imprecise information. In robotics, this expressive power has been used to design sensor interpretation systems based on qualitative,

	Grandeur	*Misère*
Behavior design	Easy heuristic coding	Poor formal tools
Behavior coordination	Logical rule format	Local minima and cycles
Environment modeling	Represent 'weak' knowledge	Unclear semantics
Layer integration	Both symbolic and numeric	—

Table 1. Main pros and cons of fuzzy logic.

approximate models of the sensors. Moreover, fuzzy set based representations can distinguish between situations of ignorance, of conflicting evidence, and of equiprobability — a distinctive feature of these representations with respect to the probabilistic ones.

While these features make fuzzy logic an attractive tool in terms of expressive power and flexibility, we should be warned that the foundations underlying the representations and operations of fuzzy logic are not fully understood yet. For instance, we still do not have a well defined operational semantics for a degree of membership or for a possibility value. Perhaps more annoyingly for practical applications, we have little theoretical guidance in the choice of the operators to be used for information aggregation and reasoning — in fact, most authors choose their operators on the ground of intuitive considerations or empirical testing. Many researchers in the field share the feeling that this is a natural situation for a discipline which is, after all, still in its puberty. Progress towards a stronger foundational theory of fuzzy logic and its semantics is a major current concern in this field (e.g., [5,16,33]).

8 Conclusions

Fuzzy logic has features that are particularly attractive in light of the problems posed by autonomous robot navigation. Fuzzy logic allows us to model different types of uncertainty and imprecision; to build robust controllers starting from heuristic and qualitative models; and to integrate symbolic reasoning and numeric computation in a natural framework. Fuzzy logic is not the philosopher's stone though, and its advantages come to a price. Table 1 summarizes the main pros and cons of fuzzy logic discussed in the previous section in the context of the four issues considered in this volume.

In closing, we can assert that fuzzy logic offers a powerful representational tool for representing "weak" types of information, including imprecise, vague, and unreliable information, at the level of detail which is available and without requiring a precise quantification of the uncertainty. Paraphrasing a motto from the domain of knowledge representation ([22], p. 53), we might say that

> The expressive power of fuzzy logic determines not so much what can be said, but what can be left unsaid.

If "strong" knowledge, like an analytical model of the controlled system or a stochastic model of the sensor, is available, then stronger techniques, like classical control or probability theory, should probably be used. If only weak knowledge is available, however, then fuzzy logic may provide a more adequate tool. Interestingly for we robot builders, much of the knowledge available in the autonomous robotics domain is inherently weak.

References

1. R. C. Arkin. *Behavior-Based Robotics.* MIT Press, Cambridge, MA, 1998.
2. J. W. Baxter and J. R. Bumby. Fuzzy logic guidance and obstacle avoidance algorithms for autonomous vehicle control. In *Proc. of the Int. Workshop on Intelligent Autonomous Vehicles*, pages 41–52, Southampton, UK, 1993.
3. H. Berenji, Y-Y. Chen, C-C. Lee, J-S. Jang, and S. Murugesan. A hierarchical approach to designing approximate reasoning-based controllers for dynamic physical systems. In *Proc. of the Conf. on Uncertainty in Artif. Intell.*, pages 362–369, Cambridge, MA, 1990.
4. H. R. Berenji. The unique strength of fuzzy logic control. *IEEE Expert*, page 9, August 1994. Response to Elkan's "The paradoxical success of fuzzy logic", same issue.
5. T. Bilgiç and I. B. Türkşen. Measurement of membership functions: theoretical and empirical work. In H.J. Zimmermann, editor, *Practical Applications of Fuzzy Technologies*, volume 6 of *Handbooks of Fuzzy Sets*. Kluwer Academic, Norwell, MA, 1999.
6. R. A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, RA-2(1):14–23, 1986.
7. M. Drummond. Situated control rules. In *Proc. of the Int. Conf. on Knowledge Representation and Reasoning*, pages 103–113, 1989.
8. E. Fabrizi, G. Oriolo, and G. Ulivi. Accurate map building via fusion of laser and ultrasonic range measures. In D. Driankov and A. Saffiotti, eds, *Fuzzy Logic Techniques for Autonomous Vehicle Navigation*, Physica-Verlag, Heidelberg, New York, 2000, pages 257–280.
9. J. Gasós. Integrating linguistic descriptions and sensor observations for the navigation of autonomous robots. In D. Driankov and A. Saffiotti, eds, *Fuzzy Logic Techniques for Autonomous Vehicle Navigation*, Physica-Verlag, Heidelberg, New York, 2000, pages 313–340.
10. E. Gat. Three-layer architectures. In R.P. Bonasso D. Kortenkamp and R. Murphy, editors, *Artificial intelligence and mobile robots*, chapter 8, pages 195–210. MIT Press, Cambridge, MA, 1998.
11. R. Ghanea-Hercock and D. P. Barnes. An evolved fuzzy reactive control system for co-operating autonomous robots. In *Proc. of the Int. Conf. on Simulation and Adaptive Behavior (SAB)*, Cap Cod, 1996.
12. J. Godjevac and N. Steele. Neuro fuzzy control for basic mobile robot behaviours. In D. Driankov and A. Saffiotti, eds, *Fuzzy Logic Techniques for Autonomous Vehicle Navigation*, Physica-Verlag, Heidelberg, New York, 2000, pages 98–118.

13. S. G. Goodridge and M. G. Kay. Multi-layered fuzzy behavior fusion for reactive control of autonomous robots. In D. Driankov and A. Saffiotti, eds, *Fuzzy Logic Techniques for Autonomous Vehicle Navigation*, Physica-Verlag, Heidelberg, New York, 2000, pages 179–204.
14. S. Hanks and R. J. Firby. Issues and architectures for planning and execution. In *Workshop on Innovative APproaches to Planning, Scheduling and Control*, San Diego, CA, 1990.
15. F. Hoffmann. The role of fuzzy logic control in evolutionary robotics. In D. Driankov and A. Saffiotti, eds, *Fuzzy Logic Techniques for Autonomous Vehicle Navigation*, Physica-Verlag, Heidelberg, New York, 2000, pages 119–148.
16. U. Höhle. A survey on the fundamentals of fuzzy set theory. In F. Kreith, editor, *The CRC Handbook of Mechanical Engineering*. CRC Press, 1998.
17. O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, 5(1):90–98, 1986.
18. H. Kiendl and J. J. Rüger. Stability analysis of fuzzy control systems using facets functions. *Fuzzy Sets and Systems*, 70:275–285, 1995.
19. K. Konolige, K.L. Myers, E.H. Ruspini, and A. Saffiotti. The Saphira architecture: A design for autonomy. *Journal of Experimental and Theoretical Artificial Intelligence*, 9(1):215–235, 1997.
20. B. J. Kuipers. The spatial semantic hierarchy. *Artificial Intelligence*, 119:191–233, 2000.
21. J. C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, 1991.
22. H. J. Levesque and R. J. Brachman. A fundamental tradeoff in knowledge representation and reasoning. In R. J. Brachman and H. J. Levesque, editors, *Readings in Knowledge Representation*, pages 41–70. Morgan Kaufmann, Los Altos, CA, 1985.
23. M. López-Sánchez, R. Lṕez de Màntaras, and C. Sierra. Map generation by cooperative autonomous robots using possibility theory. In D. Driankov and A. Saffiotti, eds, *Fuzzy Logic Techniques for Autonomous Vehicle Navigation*, Physica-Verlag, Heidelberg, New York, 2000, pages 281–312.
24. M. Maeda, Y. Maeda, and S. Murakami. Fuzzy drive control of an autonomous mobile robot. *Fuzzy Sets and Systems*, 39:195–204, 1991.
25. F. Michaud. Selecting behaviors using fuzzy logic. In *Proc. of the IEEE Int. Conf. on Fuzzy Systems*, pages 585–592, Barcelona, SP, 1997.
26. R. R. Murphy. Fuzzy logic for fusion of tactical influences on vehicle speed control. In D. Driankov and A. Saffiotti, eds, *Fuzzy Logic Techniques for Autonomous Vehicle Navigation*, Physica-Verlag, Heidelberg, New York, 2000, pages 73–98.
27. A. Ollero, J. Ferruz, O. Sanchez, and G. Heredia. Mobile robot path tracking and visual target tracking using fuzzy logic. In D. Driankov and A. Saffiotti, eds, *Fuzzy Logic Techniques for Autonomous Vehicle Navigation*, Physica-Verlag, Heidelberg, New York, 2000, pages 51–72.
28. J. Pan, D. J. Pack, A. Kosaka, and A. C. Kak. FUZZY-NAV: a vision-based robot navigation architecture using fuzzy inference for uncertainty reasoning. In *Proc. of the World Congress on Neural Networks*, pages 602–607, Washington, DC, 1995.
29. F. Pin and Y. Watanabe. Resolving conflict between behaviors using suppression and inhibition. In D. Driankov and A. Saffiotti, eds, *Fuzzy Logic Tech-*

niques for Autonomous Vehicle Navigation, Physica-Verlag, Heidelberg, New York, 2000, pages 151–178.
30. P. Pirjanian and M. Mataric. Multiple objective vs. fuzzy behavior coordination. In D. Driankov and A. Saffiotti, eds, *Fuzzy Logic Techniques for Autonomous Vehicle Navigation*, Physica-Verlag, Heidelberg, New York, 2000, pages 235–254.
31. J. K. Rosenblatt and D. W. Payton. A fine-grained alternative to the subsumption architecture for mobile robot control. In *Proc of the IEEE Int. Conf. on Neural Networks*, volume 2, pages 317–324, Washington, DC, 1989. IEEE Press.
32. E. H. Ruspini. Fuzzy logic in the Flakey robot. In *Proc. of the Int. Conf. on Fuzzy Logic and Neural Networks (IIZUKA)*, pages 767–770, Iizuka, JP, 1990.
33. E. H. Ruspini. On the semantics of fuzzy logic. *Int. J. of Approximate Reasoning*, 5:45–88, 1991.
34. E. H. Ruspini. Truth as utility: A conceptual synthesis. In *Proc. of the Conf. on Uncertainty in Artif. Intell.*, pages 316–322, Los Angeles, CA, 1991.
35. A. Saffiotti, K. Konolige, and E. H. Ruspini. A multivalued-logic approach to integrating planning and control. *Artificial Intelligence*, 76(1-2):481–526, 1995.
36. A. Saffiotti, E. H. Ruspini, and K. Konolige. Blending reactivity and goal-directedness in a fuzzy controller. In *Proc. of the IEEE Int. Conf. on Fuzzy Systems*, pages 134–139, San Francisco, California, 1993. IEEE Press.
37. A. Saffiotti and L. P. Wesley. Perception-based self-localization using fuzzy locations. In L. Dorst, M. van Lambalgen, and F. Voorbraak, editors, *Reasoning with Uncertainty in Robotics*, number 1093 in LNAI, pages 368–385. Springer-Verlag, Berlin, DE, 1996.
38. M Sugeno, M. F. Griffin, and A. Bastian. Fuzzy hierarchical control of an unmanned helicopter. In *Proc. of Int. Fuzzy System Association Conference (IFSA)*, pages 179–182, Seoul, KR, 1993.
39. M. Sugeno and M. Nishida. Fuzzy control of model car. *Fuzzy Sets and Systems*, 16:103–113, 1985.
40. H. Surmann and L. Peters. MORIA — a robot with fuzzy controlled behaviour. In D. Driankov and A. Saffiotti, eds, *Fuzzy Logic Techniques for Autonomous Vehicle Navigation*, Physica-Verlag, Heidelberg, New York, 2000, pages 343–366.
41. T. Takeuchi, Y. Nagai, and N. Enomoto. Fuzzy control of a mobile robot for obstacle avoidance. *Information Sciences*, 43:231–248, 1988.
42. E. W. Tunstel. Fuzzy-behavior synthesis, coordination, and evolution in an adaptive behavior hierarchy. In D. Driankov and A. Saffiotti, eds, *Fuzzy Logic Techniques for Autonomous Vehicle Navigation*, Physica-Verlag, Heidelberg, New York, 2000, pages 205–234.
43. C. von Altrok, B. Krause, and H. J. Zimmermann. Advanced fuzzy logic control of a model car in extreme situations. *Fuzzy Sets and Systems*, 48:41–52, 1992.
44. C. Voudouris. *Fuzzy hierarchical control for autonomous mobile robots.* MSc Thesis, Dept. of Computer Science, Univ. of Essex, Esssex, UK, Sept 1993.
45. L.-X. Wang. A mathematical formulation of hierarchical systems using fuzzy logic systems. In *Proc. of the IEEE Int. Conf. on Fuzzy Systems*, pages 183–188, Orlando, FL, 1994.
46. J. Yen and N. Pfluger. A fuzzy logic based extension to Payton and Rosenblatt's command fusion method for mobile robot navigation. *IEEE Trans. on Systems, Man, and Cybernetics*, 25(6):971–978, 1995.

47. L. A. Zadeh. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, 1:3–28, 1978.
48. J. Zhang and A. Knoll. Integrating deliberative and reactive strategies via fuzzy modular control. In D. Driankov and A. Saffiotti, eds, *Fuzzy Logic Techniques for Autonomous Vehicle Navigation*, Physica-Verlag, Heidelberg, New York, 2000, pages 367–387.